Linux Kernel Testing: Providing Better Results to the Upstream Community

Laura Nao, Collabora Ltd

EOSS 2024

COLLABORA

Agenda

- Motivation
- Test Quality
- Device Probe Kselftests & Friends
- Test Integration and Next Steps

COLLABORA

Motivation

Test Quality Matters

- Key questions
  - How can CIs better support maintainers and developers?
  - How can we deliver higher-quality test results to the upstream community?
  - How can we emphasize the importance of automated testing?

Insights from KernelCI

- Background
  - KernelCI: https://kernelci.org/
  - Issues identified when adding new tests and monitoring results
  - User feedback on reports
  - New system on the way
- Goal
  - Foster upstream community participation
- Prerequisite
  - Restoring trust in KernelCI

Pain Points

- Unreliable results
  - Misleading reports
  - False negatives
    - Unmaintained tests
    - Usage of unstable ABI
    - Case example: bootrr
- Fragmentation
  - Seeking a common ground

COLLABORA

Test Quality

Test Quality

- Quality criteria
  - Coverage

Test Quality

- Quality criteria
  - Coverage
  - Speed/efficiency

Test Quality

- Quality criteria
  - Coverage
  - Speed/efficiency
  - Portability

Test Quality

- Quality criteria
  - Coverage
  - Speed/efficiency
  - Portability
  - Consistency

Test Quality

- Quality criteria
  - Coverage
  - Speed/efficiency
  - Portability
  - Consistency
  - Maintainability

Test Quality

- Quality criteria
  - Coverage
  - Speed/efficiency
  - Portability
  - Consistency
  - Maintainability
  - Output format compliance

Test Quality

- Quality criteria
  - Coverage
  - Speed/efficiency
  - Portability
  - Consistency
  - Maintainability
  - Output format compliance
  - Community adoption

Test Quality

- The plan
  - Focus on quality more than quantity
  - Prioritize in-tree tests
    - Collaborate with the upstream community
    - Promote reusability across different CIs
  - Make the kernel easier to test

KSelftests

- Kselftests as a starting point
  - Already enabled in KernelCI
    - Analyze failures and regressions reported
    - Analyze the output format
  - Chance to increase coverage
    - Convert unstable tests in KCI to Kselftests

COLLABORA

Device Probe Kselftests

Device Probe Tests

- Replacements for bootrr checks in KernelCI
  - Use stable ABI
  - Based on dynamic detection of the on-board peripherals (where possible)
- New kselftests to identify unprobed devices
  1. DT Kselftest (merged)
  2. ACPI Kselftest (WIP)
  3. Discoverable devices probe Kselftest (merged)

1. DT Probe Kselftest

- Goal
  - Detect unprobed devices defined in the device tree
- Process
  - Rely on two lists
    - List of `compatibles` matching DT nodes to drivers
    - Ignore list to filter out specific `compatibles`
  - Iterate through the platform DT nodes
  - Compare `compatibles` against the lists
  - Verify driver binding for each device

1. DT Probe Kselftest

- Refs
  - Available in `tools/testing/selftests/dt` since v6.6
  - `make TARGETS="dt" kselftest`
  - [PATCH v3 0/3] Add a test to catch unprobed Devicetree devices

2. ACPI Probe Kselftest

- Goal
  - Detect unprobed devices on ACPI platforms
- Process
  - Rely on two lists
    - List of ACPI IDs matching devices to drivers
    - Ignore list to filter out specific IDs
  - Iterate through ACPI objects in /sys/devices/LNXSYSTM:00
  - Compare ID against the lists
  - Verify driver binding for each physical device linked to each ACPI object

2. ACPI Probe Kselftest

- Devices declared in the ACPI namespace
  - Platform devices
    - HID/CID
    - Discovered through the platform firmware
  - Discoverable devices
    - ADR
    - Discovered and enumerated natively through bus protocols
- ACPI glue layer
    - ACPI object ↔ physical device (`physical_node*/firmware_node`)

2. ACPI Probe Kselftest

- Exceptions
  - Devices that do not require a driver
  - Devices not assigned to any subsystem
  - Devices that are linked to other devices
  - Class devices

2. ACPI Probe Kselftest

- Exceptions
  - Devices that do not require a driver → ignore list/sysfs attributes
  - Devices not assigned to any subsystem → sysfs attributes
  - Devices that are linked to other devices → sysfs attributes
  - Class devices → sysfs attributes

2. ACPI Probe Kselftest

- Refs
  - Open RFC: [RFC PATCH v2 0/2] Add a test to verify device probing on ACPI platforms

3. Discoverable Devices Probe Kselftest

- Goal
  - Detect unprobed devices on discoverable buses
- Process
  - Relies on per-platform binding describing the system topology (out of tree)
  - Verifies that the devices specified in the file are created and bound to a driver
- Refs
  - Available in `tools/testing/selftests/devices` since v6.8
  - [PATCH v4 0/3] Add test to verify probe of devices from discoverable buses
  - Detecting device probe regressions @ LPC 2023

Some Takeaways

- Bonus pain point
  - No standardized interface to verify device probing status
  - Refs
    - [RFC PATCH 0/2] Add a test to verify device probing on ACPI platforms
- Regressions/failures found
  - [PATCH 0/2] Enable PSTORE_RAM as a module in the arm64 defconfig
  - [PATCH v3] arm64: dts: mt8183: Move CrosEC base detection node to kukui-based DTs
  - Probe regression of efuse@11f10000 on mt8183-kukui-jacuzzi-juniper-sku16 running next-20240202

COLLABORA

Beyond Device Probe Tests

Related Work

- Fix KTAP output conformance and missing configs
  - [PATCH] mm/selftests: hugepage-mremap: conform test to TAP format output
  - [PATCH v4 7/7] selftests/mm: config: add missing configs
- Power supply properties conformance
  - [PATCH] selftests: Add test to verify power supply properties
- Rust sample modules probe
  - [PATCH v5] kselftest: Add basic test for probing the rust sample modules

Related Work

- WIP
  - Suspend/resume test (standalone/cpufreq Kselftest)
  - Error log parsing
    - Make kernel errors more reliable
    - [PATCH 0/3] device: core: Adjust device probe log messages to ease error detect ion
- Out-of-tree tests
  - Fluster+GStreamer V4L2 decoder conformance
  - Watchdog reset functionality

COLLABORA

Test Integration and Next Steps

Test Integration

- New KernelCI system
  - https://kernelci.org/blog/posts/2024/strategic-updates/
  - Community engagement working group
    - https://lore.kernel.org/kernelci/bf81be70-61ec-4169-b66a-5c3136869107@gmail.com/T/#u
  - Rebuilt from the ground up
  - Gradually introducing new tests, fine combing the results
  - More awareness on quality and long term maintainability

COLLABORA

Recap

- Motivation
- Test Quality
- Device Probe Kselftests & Friends
- Test Integration and Next Steps

More on Kernel Testing

- KernelCI - How Can It Impact Your Future? - Paweł Wieczorek, Collabora
- Quickly Test Your Kernel with GitLab CI - Helen Koike, Collabora
- Join us on Tuesday April 16th at 7pm @ Elysian Capitol Hill Brewery
  - 1221 E Pike St, Seattle (WA) - https://maps.app.goo.gl/K6G2XMVpHHzKjvV79
  - https://lore.kernel.org/kernelci/2df803a1-aa81-0762-183a-0aefa527c587@collabora.com/T/#t

Thank you!

We are hiring

col.la/careers